Evaluation of Spatial Keyword Queries with Partial Result Support on Spatial Networks

Ji Zhang[1], Wei-Shinn Ku[1], Xunfei Jiang[1], Xiao Qin[1], Yu-Ling Hsueh[2]
[1]Dept. of Computer Science and Software Engineering, Auburn University, Auburn, AL, USA
[2]Dept. of Computer Science and Information Engineering, National Chung Cheng University, Chia-yi, Taiwan
Email: {jizhang, weishinn, xunfei, xqin}@auburn.edu, hsueh@cs.ccu.edu.tw

***Abstract*—Numerous geographic information system applications need to retrieve spatial objects which bear user specified keywords close to a given location. In this research, we present efficient approaches to answer spatial keyword queries on spatial networks. In particular, we formally introduce definitions of Spatial Keyword k Nearest Neighbor (SKkNN) and Spatial Keyword Range (SKR) queries. Then, we present a framework of a spatial keyword query evaluation system which is comprised of Keyword Constraint Filter (KCF), Keyword and Spatial Refinement (KSR), and the spatial keyword ranker. KCF employs an inverted index to calculate keyword relevancy of spatial objects, and KSR refines intermediate results by considering both spatial and keyword constraints with the spatial keyword ranker. In addition, we design novel algorithms for evaluating SKkNN and SKR queries. These algorithms employ the inverted index technique, shortest path search algorithms, and network Voronoi diagrams. Our extensive simulations show that the proposed SKkNN and SKR algorithms can answer spatial keyword queries effectively and efficiently.**

I. Introduction

A Spatial Keyword (SK) query is an approach for searching qualified spatial objects by considering both the query requester's location and user specified keywords. Taking both spatial and keyword requirements into account, the goal of a spatial keyword query is to efficiently find results that satisfy all the conditions of a search. However, most existing solutions for SK queries are designed based on Euclidean distance [2], [4], [13], [11], which is not realistic since most users move on spatial networks. Moreover, most current approaches for SK queries are limited to finding objects that fully match the given keywords. Nevertheless, the objects with fully matched keywords could be *far away* from the query point. In this research, we design novel SK query techniques based on spatial networks. In addition, we take both fully and partially matched query results into account in the process of keyword searching. This new SK query mechanism enables users to not only retrieve qualified results on spatial networks, but also obtain partially matched objects when there are not enough fully matched results *in the vicinity* of the requester.

Figure 1 illustrates an example: a tourist who flies to Atlanta may want to search for two hotels which provide both "Internet" and "Breakfast" amenities and have the shortest driving distance to the Atlanta airport. In addition, the tourist may also search for all the hotels which are within 10 miles of the airport and provide the two amenities in order to compare the hotels' reviews and prices. For retrieving the qualified

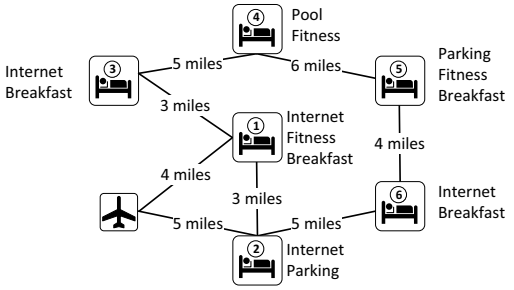

Fig. 1. A sample spatial network of hotels close to an airport.

hotels, the tourist will launch a Spatial Keyword k Nearest Neighbor (SKkNN) query with ranking parameters for the first search; the query results are hotels 1 and 3. A Spatial Keyword Range (SKR) query will be executed for the second inquiry, and the answers are hotels 1, 3, and 6. In this paper, we focus on solving the two aforementioned spatial query types by devising three novel solutions which employ the inverted index technique, shortest path search algorithms, and network Voronoi diagrams. Particularly, the inverted index is used to maintain the relationships between spatial objects and their attached keywords for quickly retrieving spatial objects whose features match the given keywords. In addition, we propose a network expansion-based approach and a Voronoi diagram-based approach to efficiently answer SKkNN queries on spatial networks. The contributions of this study are as follows:

1) We provide formal definitions of spatial keyword kNN and range queries on spatial networks.
2) We develop two novel approaches for efficiently processing SKkNN query and one approach for SKR query on spatial networks.
3) Our SKkNN solution can return partially matched query results based on the output of the spatial keyword ranker.
4) We evaluate the performance of the proposed SKkNN and SKR algorithms through extensive experiments with both real-world and synthetic data sets.

The rest of this paper is organized as follows. The proposed query types are formally defined in Section II. In Section III, we introduce the spatial keyword query evaluation algorithms. Due to the space limitation, the experimental results can be found in [12]. Section IV concludes the paper with a discussion of future work.

978-0-7695-4973-6/13 $26.00 © 2013 IEEE
DOI 10.1109/MDM.2013.41

II. Query Type Definition and Background

A. Foundation

In this subsection, we introduce the foundation of spatial keyword queries. In an SK query, a spatial object p is defined as a pair $<l, t>$, where l is a location in the search space and t is a text description (e.g., amenities and features of a hotel) of the corresponding object. Table I summarizes notations used in this paper.

1) Distance on Spatial Networks: Spatial networks are composed of undirected weighted graphs $G = (V, E)$, where V is a set of vertices and E is a set of edges. In general, the weight of each edge is determined by a metric measured in physical distance or time cost for traveling the road segment [6], [7]. The distance between two objects $D_n(.,.)$ on spatial networks is the summation of all segment weights on the shortest path connecting the two objects.

2) Matched Keywords: Matched-keywords is a set of keywords which are in both sets of $p.t$ and K, where $p.t$ is the text description of a given spatial object, and K is a set of keywords specified by a user.

$$MK(p, K) = \{\, k_i \in K \mid k_i \in p.t \,\} \tag{1}$$

3) Fully Matched Keyword Search: With a given data set, the purpose of Fully Matched Keyword Search (FMKS) is to find objects whose descriptions completely match with a set of keywords K specified by a requester. As shown in Equation (2), the descriptions of search results of FMKS may be either identical to K or a superset of K.

$$FMKS(P, K) = \{\, p_i \in P \mid K \subseteq p_i.t \,\} \tag{2}$$

4) Partially Matched Keyword Search: With a given data set, the purpose of Partially Matched Keyword Search (PMKS) is to retrieve objects which match at least one keyword in the user defined keyword set as shown in Equation (3).

$$PMKS(P, K) = \{\, p_i \in P \mid \exists k_j \in p_i.t \ and \ k_j \in K \,\} \tag{3}$$

5) Weighted Keyword Relevancy: We use a weight function TR to calculate keyword relevancy of a specific spatial object p [10]. We assume that each keyword k_i in a keyword set K is assigned with a weight $w(k_i)$, which indicates its importance in queries. Consequently, given an object p and a keyword set K, we have the following equation:

$$TR(p, K) = \sum_{k_i \in MK(p,K)} w(k_i) \tag{4}$$

For special cases where all keywords share identical weight, Equation (5) can be derived from Equation (4) where $w(k_i) = 1$ and $|MK(p, K)|$ is the number of keywords in $MK(p, K)$.

$$TR(p, K) = \sum_{k_i \in MK(p,K)} 1 = |MK(p, K)| \tag{5}$$

TABLE I Symbolic notations.

Symbol	Meaning
P	A set of spatial objects
Q	A spatial keyword query
K	A set of search keywords
q	The location of a requester
k	The requested number of objects in the result of a SKkNN query
r	The search range of a SKR query
s	The ranking score of an object
$\|\mathbb{S}\|$	The number of elements in set $\mathbb{S}$
$d(.,.)$	The Euclidean distance between two points
$D_n(.,.)$	The shortest network distance between two points
$\mathbb{R}$	The result set of a query
$\mathbb{E}$	The explored region of a VDkNN query

B. Spatial Keyword Ranker

A spatial keyword ranker is designed to determine the ranking of a given spatial object in a SKkNN query by employing both metrics, spatial network distance and keyword relevancy. We utilize a ranking function RK to compute how well an object matches an SKkNN query. Given a query $Q <l$, $K>$ and an object $p <l$, $t>$, the ranking function is defined as follows:

$$RK(Q, p) = \theta_1 \cdot TR(p.t, Q.K) - \theta_2 \cdot D_n(p.l, Q.l) \tag{6}$$

In Equation (6), θ_1 and θ_2 are parameters of each part of the function [4], and their values depend on user preferences. For example, if a user is more concerned about keyword match, θ_1 can be set to a larger value than θ_2 in order to make keyword relevancy dominant in the RK function. Moreover, intuitively, an object with either a shorter distance or a higher keyword relevancy would have a higher ranking in query results. Therefore, TR has a positive influence on the RK function while D_n has a negative one.

C. Spatial Keyword kNN Queries

Based on the spatial keyword ranker, the purpose of a spatial keyword kNN query is to retrieve k objects which have top k ranking values.

Definition Given an SKkNN query Q and an object set P, we define SKkNN(P, Q, k) as follows:

$$RK(p_i) \geq RK(p_j) \ where \ p_i \in SKkNN(P, Q, k) \wedge p_j \in P \backslash \{SKkNN(P, Q, k)\} \wedge |SKkNN(P, Q, k)| = k \tag{7}$$

D. Spatial Keyword Range Queries

An SK Range query finds all the objects that fully match the given keywords within a user specified distance.

Definition Let P be a set of objects. Given a query location q, a search range r, and a set of keywords K, an SK range query is defined as follows:

$$SKR(P, q, r, K) = \{p_i \in P | K \subseteq p_i.t \wedge D_n(p_i, q) \leq r\} \tag{8}$$

III. System Design

In this section, we design a spatial keyword query evaluation system which is comprised of Keyword Constraint Filter (KCF), Keyword and Spatial Refinement (KSR), and the spatial keyword ranker.

A. Framework of Query Evaluation

Before presenting the details of our spatial keyword query algorithms, we briefly introduce the framework of our system. As illustrated in Figure 2, the spatial keyword query evaluation system comprises three main components. The system receives both spatial data sets and spatial keyword constraints as inputs and produces results after a two-step computation.

A filter-and-refine strategy is employed to answer SK queries. The two key steps are KCF and KSR. KCF receives spatial data sets and keyword constraints and filters out objects that do not match any user specified keyword. Because spatial network distance computation is expensive, we do not take spatial constraints into account in this step. The main purpose of KCF is to reduce the number of candidate objects in order to decrease computation costs in the next step. In the second step, KSR receives inputs from KCF and refines the intermediate results based on both keyword and spatial constraints. Afterward, KSR returns the qualified objects sorted by their ranking scores provided by the ranker.

B. Keyword Constraint Filter

1) Inverted Indexing Structure: Inverted indexes are primarily designed to support keyword searches from a set of text files [8]. In our system, we utilize inverted indexes to search for objects related to specific keywords from spatial databases. An index of terms is maintained in our system where each term is a unique keyword, and each postings list contains a number of object identifiers. Each postings list is in sorted order (based on object identifiers) to facilitate the efficient search of objects related to a specific keyword. If an object has multiple keywords, its identifier will appear in each corresponding postings list. In addition, inverted indexes are independent of other dedicated index structures, such as R-trees or grids, in spatial databases.

2) Keyword Match Algorithm: Based on the proposed problem, we utilize a keyword match algorithm by employing the inverted index-based merge technique [8] to calculate the keyword relevancy of spatial objects. With the keyword match algorithm, we measure the keyword relevancy of a spatial object by counting the number of matched-keywords. The more matched-keywords an object has, the higher its keyword relevancy is. This algorithm receives an inverted index and a set of keywords as input parameters and then returns the keyword relevancy of objects that match with at least one keyword.

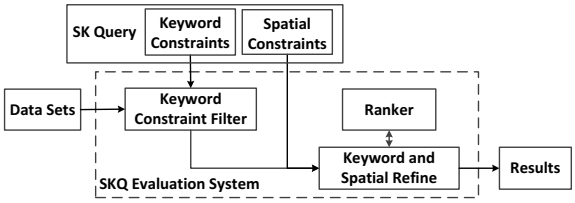

Fig. 2. Framework of the proposed system.

C. Network Expansion-Based SKkNN Query Algorithm

In this section, we explain our algorithm for processing spatial keyword k nearest neighbor queries based on network expansion techniques [1], [9]. The algorithm receives an inverted index, a query point q, the value of k, and a set of keywords K as input parameters and returns the top k objects by considering both keyword and spatial constraints.

For searching the shortest path between objects on spatial networks, Dijkstra's algorithm-based approaches [1], [3] have been widely utilized in various applications. Given a source point and a group of destinations, the algorithm recursively expands the unvisited paths and records distances of intermediate nodes. During the search, a distance record of a node will be updated if there is a shorter path than the present one. Such a process is continued until all the destinations have arrived and the distances of all other possible paths are longer than their current distances. In addition, a solution named Incremental Network Expansion (INE) is presented in [9] by extending Dijkstra's algorithm to compute k nearest neighbors in a network space. Specifically, INE first locates the network segment e_i, which covers the query point q, and retrieves all objects on e_i. If any object p_i is found on e_i, p_i will be inserted into the result set. Furthermore, the endpoint of e_i, which is closer to q, will be expanded while the second endpoint of e_i will be placed in a priority queue Q_p. INE repeats the process by iteratively expanding the first node in Q_p and inserting newly discovered nodes into Q_p until k objects are retrieved.

We develop a Network Expansion-based SKkNN (NEkNN) solution by leveraging INE. There are two main steps in the NEkNN algorithm. The first step is to filter out objects which do not match any user specified keywords by employing methods discussed in Section III-B. Then, we mark all the remaining objects in the spatial network as candidates (e.g., set a bit of these points of interest). The next step is to expand the network from q with INE and the ranking function (Section II-B). When an object p_i is discovered, NEkNN verifies that p_i is a candidate object. If p_i is a candidate object, NEkNN calculates its ranking score s by executing the ranking function (otherwise the algorithm ignores p_i). Meanwhile, NEkNN keeps a result set $\mathbb{R}$ which is sorted in descending order based on the ranking score. If $\mathbb{R}$ has fewer than k objects and p_i is a candidate object, p_i is inserted into $\mathbb{R}$. Otherwise, NEkNN compares the ranking score of p_i with the last object p_j in $\mathbb{R}$. p_j will be replaced by p_i if $p_i.s > p_j.s$.

In addition, when $|\mathbb{R}| \geq k$, NEkNN calculates ranking scores for network nodes as well by assuming that they match all the search keywords to restrict the search space. In other words, any spatial object p_i, which is further away from q than a network node n_i, must have a lower ranking score than n_i even if p_i matches all the search keywords. Consequently, NEkNN iterates the search process until $\mathbb{R}$ contains k objects and the next network node to be expanded in Q_p has an equal or lower ranking score than the last object in $\mathbb{R}$.

D. Voronoi Diagram-Based SKkNN Query Algorithm

Although NEkNN is able to restrict the search space and retrieve the top k objects based on their ranking scores, the main limitation of NEkNN is that it has to explore a large portion of the network when candidate objects are not densely distributed in the network. Therefore, we propose a Voronoi diagram-based SKkNN (VDkNN) solution by leveraging the network Voronoi diagram (NVD) [5] to improve performance. In order to be independent of the density and distribution of candidate objects, we first partition the spatial network into small regions by generating a network Voronoi diagram over all the spatial objects (points of interest). Each cell of the NVD is centered on one spatial object and contains the nodes that are closest to the object based on network distance. Afterward, for each NVD cell, we pre-compute the distances between all the edges of the cell to its center as well as the distances across the border points of the adjacent cells. Consequently, for a new cell, we can quickly extend the region to the border points without expanding the internal network segments.

With the NVD of the search space, for an SKkNN query, VDkNN first filters out unqualified objects with methods discussed in III-B and marks all the remaining objects in the NVD as candidates. Then, VDkNN finds the network Voronoi polygon NVP(p_i) that contains q where p_i is the generator of the polygon. This step can be accomplished by employing a spatial index (e.g., the R-tree), which is generated based on the NVD cells. Next, we verify that p_i is a candidate object. If p_i is a candidate object, VDkNN calculates its ranking score by running the ranking function (Section II-B). In addition, VDkNN maintains a result set $\mathbb{R}$ which is sorted in descending order according to the ranking score. When $\mathbb{R}$ contains fewer than k objects, newly discovered candidate objects are inserted into $\mathbb{R}$. However, if $\mathbb{R}$ already includes k objects, VDkNN replaces the k^{th} object p_k of $\mathbb{R}$ when a newly retrieved candidate object has a higher score than p_k. Also, VDkNN keeps a queue Q_n which stores the neighbors (adjacent cells) of p_i and a set $\mathbb{E}$ which consists of all the searched cells (i.e., $\mathbb{E}$ covers the current explored region).

Subsequently, VDkNN searches the adjacent cells of $\mathbb{E}$ (i.e., NVP(p_i)) stored in Q_n for the next candidate object. Every time after a cell NVP(p_j) been explored, the neighboring generators of p_j are unioned with Q_n, NVP(p_j) is unioned with $\mathbb{E}$, and $\mathbb{R}$ is updated according to the aforementioned rules if p_j is a candidate object. Moreover, when $|\mathbb{R}| \geq k$, VDkNN calculates the ranking score of all the border points of the current explored region by assuming that they match all the search keywords to restrict the search space. VDkNN iterates the search process until $\mathbb{R}$ contains k objects and the ranking scores of all the border points of $\mathbb{E}$ are equal or worse than the ranking score of the k^{th} object in $\mathbb{R}$ (i.e., there will not be any changes in $\mathbb{R}$ even if we search further).

E. Spatial Keyword Range Query Algorithm

As defined in Section II-D, given a query point q, a search range r and a set of keywords K, SKR query retrieves all the objects which fully match all the keywords within r. SKR query first calculates the keyword relevancy of objects. Then, it retrieves objects which fully match all the given keywords and stores the qualified objects in $\mathbb{R}$. Afterward, it calls Dijkstra's algorithm for calculating distances from q to all the candidate objects. Finally, SKR query removes objects which are out of the search range from $\mathbb{R}$.

F. Experimental Validation

The worst-case running time of our NEkNN and SKR approaches on a spatial network with a set of nodes N is $O(|K| * |P| + |N|^2)$ by considering both the keyword match and spatial network search subroutines. We evaluate the performance of our solutions with both real-world and synthetic data sets. Due to the space limitation, all experimental results can be found in [12].

IV. Conclusion

Geographic information systems are becoming increasingly sophisticated, and spatial keyword search represents an important class of queries. Most existing solutions for evaluating spatial keyword queries are based on Euclidean distance and cannot provide partially matched results. In this research, we introduce efficient techniques to answer spatial keyword k nearest neighbor and spatial keyword range queries on spatial networks. We demonstrate the excellent performance of the proposed algorithms through extensive simulations. For future work, we plan to extend our spatial keyword query evaluation framework to support other common spatial query types such as spatial join, reverse nearest neighbor, spatial skyline, etc.

References

[1] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[2] I. D. Felipe, V. Hristidis, and N. Rishe. Keyword Search on Spatial Databases. In *ICDE*, pages 656–665, 2008.

[3] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34(3):596–615, 1987.

[4] R. Hariharan, B. Hore, C. Li, and S. Mehrotra. Processing Spatial-Keyword (SK) Queries in Geographic Information Retrieval (GIR) Systems. In *SSDBM*, page 16, 2007.

[5] M. R. Kolahdouzan and C. Shahabi. Voronoi-Based K Nearest Neighbor Search for Spatial Network Databases. In *VLDB*, pages 840–851, 2004.

[6] W.-S. Ku, R. Zimmermann, H. Wang, and T. Nguyen. Annatto: Adaptive nearest neighbor queries in travel time networks. In *MDM*, page 50, 2006.

[7] W.-S. Ku, R. Zimmermann, H. Wang, and C.-N. Wan. Adaptive nearest neighbor queries in travel time networks. In *GIS*, pages 210–219, 2005.

[8] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[9] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao. Query Processing in Spatial Network Databases. In *VLDB*, pages 802–813, 2003.

[10] D. Wu, M. L. Yiu, C. S. Jensen, and G. Cong. Efficient Continuously Moving Top-K Spatial Keyword Query Processing. In *ICDE*, 2011.

[11] D. Zhang, Y. M. Chee, A. Mondal, A. K. H. Tung, and M. Kitsuregawa. Keyword Search in Spatial Databases: Towards Searching by Document. In *ICDE*, pages 688–699, 2009.

[12] J. Zhang, W.-S. Ku, and X. Qin. Spatial Keyword Queries with Partial Support on Spatial Networks. Technical Report CSSE13-01, February, 2013. http://www.eng.auburn.edu/files/acad_depts/csse/csse_technical_reports/csse13-01.pdf.

[13] Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma. Hybrid Index Structures for Location-based Web Search. In *CIKM*, pages 155–162, 2005.